Metacomputing: What do Users Want?

David Cronk and Shirley Moore
Computer Science Department
University of Tennessee, Knoxville

Metacomputing, or grid computing, continues to grow, both in terms of popularity as well as functionality. While large grid systems, such as Globus, continue to generate the most publicity, many other systems are emerging as well. Some of these systems are layered on top of existing systems such as Globus, while others are stand alone systems. The common goal of all these systems is to seamlessly tie together a group of geographically distributed HPC resources into a single, shared resource. The differences, however, can be substantial, both in terms of functionality and in terms of ease of use and maintenance. In order to leverage these emerging technologies, the DoD must evaluate the cost effectiveness of various solutions as well as decide on a target group of users.

The authors' opinion is that there are basically three types of domain scientists doing DoD work. The first group consists of scientists who are not computer savvy. These scientists do not currently do any High Performance Computing (HPC) and do little or no programming. They make heavy use of libraries and often use interactive tools such as Matlab. The second group consists of moderately computer savvy domain scientists. These scientists do some coding and algorithm development and generally understand how code can be optimized for performance. While they may do some HPC, they rarely parallelize the code themselves. These scientists are mainly concerned with submitting their jobs and getting results. The third group of scientists consists of sophisticated HPC users. These scientists embrace new technologies and are willing to put forth significant effort to reduce response time for results.

Each of these different types of users has different requirements that must be met before any effort will be put forth to use any type of grid computing environment. These requirements are in terms of both ease of use and benefits to be gained. This paper studies these different types of users in an attempt to determine their needs. These studies are conducted through a combination of questionnaires and one on one interaction with current DoD users. It is the authors' opinion that the DoD cannot effectively choose new technologies to support until it has a clear understanding of its users' needs and desires, as well as the cost of maintaining and supporting different systems. This paper is a first step in this understanding. The authors feel this is very important since preliminary results obtained through a survey of DoD users shows very little current interest in grid computing. This study should help both in selecting appropriate projects/systems to fund and which users to target and how.

1. INTRODUCTION

Metacomputing, or grid computing, continues to grow, both in terms of popularity as well as functionality. While large grid systems, such as Globus, continue to generate the most publicity, many other systems are emerging as well. These systems range in sophistication from very simple to extremely complex. While it is clear some very exciting new technology is being developed, the people involved in the grid community must be sure not to lose sight of for whom this work is being done. While there are many computer scientists and managers very excited about metacomputing, few end users are involved in the movement. For the purpose of this paper, end users refer to domain scientists. Computer scientists have a history of developing new interesting technologies, handing this technology over to end users, and saying, “here, this will solve all your problems”. Unfortunately, this is too often done with very little prior interaction with said end users to determine just what their problems and needs are.

The DoD would benefit from getting more input from the domain scientists to help guide what types of projects are useful to the DoD, and therefore worthy of funding. The DoD also may benefit from determining what types of users are to be targeted for grid systems. Users who have had little or no exposure to High Performance Computing (HPC) will only benefit from certain types of systems, while sophisticated HPC users will only benefit from very different systems. While ease of use may be the most important condition for the former, tangible benefits may be the most important consideration to the latter group.

Once the target users are determined, their actual needs can be studied. It makes little sense to spend time and money developing a system no one will actually use. One must remember that the end users are mostly interested in their science, and not in new and interesting ways to do things. Unless a new system is going to offer them benefits that out weigh the cost of learning the new system, most users are not going to use said new system. A crucial step to providing systems that meet the users' needs is to understand these needs.

The rest of this paper is organized as follows: Section 2 presents an overview of metacomputing with section 3 covering a rough classification of different types of end users, in terms of HPC experience. Section 4 presents the results of surveys and one-on-one discussions with users about what they want from a metacomputing system. Finally, section 5 offers some conclusions based on these user needs.

2. METACOMPUTING

The term *metacomputer* typically denotes a networked virtual computer, consisting of possibly geographically distributed resources connected by high-speed networks. Metacomputing is motivated by a need to access computing resources not often located within a single computing system. These resources are often not co-located for a variety of reasons ranging from the cost of supercomputers to the infrequency of needing certain configurations [11].

Metacomputing [12,13] systems allow large-scale applications to make use of collections of high performance computing resources in a seamless manner [10]. These systems hide the complexity of managing such a system from the user, allowing the user to be more concerned with the science, and less concerned with implementation details. That is to say, the user can be working from a desktop machine while the underlying system handles the details of accessing remote resources and coordinating the computation. Such a system provides many potential benefits to the user. One very important such benefit is that the user need not learn the sometimes cryptic login and job submission protocols for each high-performance system to which he has access.

Another benefit is that with a metacomputing system a single problem may execute on multiple supercomputers simultaneously. This allows problems that are too large to execute on traditional high performance systems to execute in a larger *metasystem*. A further benefit of this is that certain applications can take advantage of different architectures for different tasks in the problem. An example might be a simulation problem that uses a particular architecture to perform the simulation, while another architecture is used to visualize the results. This division of tasks on different architectures may even take place without the user's knowledge. This helps the user to make better use of the resources available. A well-designed metacomputing system will allow the user to submit a job once, and the software will select the appropriate resources on which to execute the different tasks. This resource selection should be based on many factors, including current load of the resource and the appropriateness of the resource for the type of task.

While running a single job on multiple resources has been shown to be possible in very strictly controlled experiments, the reality of doing this in a production environment seems years away. Thus, for the foreseeable future, metacomputing should practically refer to seamlessly providing access to geographically distributed resources. Different systems succeed at providing this seamless access to different degrees.

There are large, monolithic systems, which provide most of the functionality of a grid system [11,15,16,19,28]. Popular examples of such middleware systems are Globus [11,16] and Legion [19]. However, these systems tend to offer little in terms of hiding the complexity of the grid from the users. Thus, the majority of domain scientists have little or no interest in learning these systems. This has led to many projects that either build on top of existing grid middleware, such as grid portals [17,18], or systems that provide abstractions for building grid applications [7,23]. Often these systems which provide abstractions, are themselves layered on top of grid middleware. Another possible solution is to build integrated software environments. These environments may be general purpose [4,21] or targeted to specific applications [8,20,24,25].

Of systems that provide abstractions for building grid applications, network-enabled server (NES) systems [9] are becoming popular. Two of the most successful of this type of system are NetSolve [1,7,22] and Ninf [23,28]. Network-enabled servers provide users access to both hardware and software resources, distributed across a network. This is done via an RPC type mechanism. This allows scientists and engineers to make use of libraries with no need to locate, configure, compile, install, or upgrade these libraries [22].

NES systems are supposed to hide all the details of resource acquisition, executable locations, and data transfer. The user simply sees a wealth of scientific software resources available, explicitly calls a program or routine, and the NES system handles resource selection, execution, data management, and the return of results [9]. This is often done from a familiar interface such as Matlab or Mathmatica, freeing the user from learning new interfaces.

One of the most common obstacles to efficient program execution in a grid system is data management [3]. Many applications that could benefit from a grid system, deal with very large datasets. From a users standpoint, an ideal grid system should allow simple specification of data files, and the actual movement of data (file staging) should be handled automatically by the system. There is significant ongoing research into the data management issues [2,6,20,26].

Another common obstacle to efficient program execution is scheduling. When dealing with many hardware resources, possible geographically distributed across large distances, difficult scheduling issues arrive. Additionally, the best solution to many of these issued depends upon the desired result. While one mechanism may provide highest resource utilization, another may provide fastest average time to completion. The desired results play an essential role in developing scheduling solutions. Significant work continues to be applied to the many scheduling issues [5,14,29,30,31].

While all these issues are very important, and certainly difficult to address, they must be addressed in a manner that hides details from the end-user in order for grid computing to be successful.

3. USERS

DoD domain scientists can be roughly classified in three groups of users. The first group of users consists of scientists who have little or no computer savvy. The second group of users consists of scientists who have computer savvy, but are not sophisticated users of the resources available. The final group of users consists of scientists who are sophisticated users of high performance computing (HPC) resources.

3.1 Unsophisticated Users

There is a large group of DoD domain scientists who know very little about the computers they are using and view these machines simply as black boxes that generate results to be analyzed. These users do little or no programming, often working with codes written and maintained by others. When programming is performed, these users generally make heavy use of numerical libraries and/or interactive environments such as Matlab or Mathmatica. They typically do not do any algorithm development of their own. These users prefer to use tools that allow them to submit data with the tools automatically generating results. Little understanding of the actual computation method is required.

Clearly, these users are not doing any high performance computing. While this may seem to make them poor candidates for users of a grid system, the exact opposite may be true. If a metacomputing system can offer improved performance with little effort on the user's part, this group of users may be the perfect target for a grid system.

3.2 Intermediate Users

The group of intermediate users is a very dynamic group. As unsophisticated users decide to learn HPC and what technology has to offer, they begin to join this group. As members of this group explore more and more opportunities in HPC, they begin to join the group of sophisticated users.

Intermediate users do some programming and algorithm development. They understand how the machines work and may perform some levels of code optimization. While this group does a lot of high performance computing, they are generally working with codes that were parallelized by others. This group does very little hand parallelization or optimization of parallel code. In general, this group understands how to submit jobs on some of the HPC machines and how to collect results, but the effort and interest end there.

This may be an excellent group to target with grid systems. These users are generally characterized by interest in better performance, but with more interest in their science than in learning more about the computers their codes are running on.

3.3 Sophisticated Users

The group of sophisticated users is the smallest of the three presented here. These users truly embrace new technologies and put forth significant effort to improve the performance of their codes. These users often hand parallelize sequential code and/or optimize parallel algorithms. These users have deep knowledge of how their computations are being performed and understand the different interactions between different parts of the HPC resources being utilized. It is not unusual for these users to understand the intricacies of different architectures and to, in fact, optimize their code for specific architectures.

On the surface, this group may appear to be the most receptive to utilizing a grid system. However, it is important to remember that, while these users are willing to put forth significant effort for improved performance, they are very results oriented. These users often are uninterested in fads that will offer no tangible benefits.

4. RESULTS

Results for this paper were gathered through email surveys and face-to-face conversations with users. Users were asked about their understanding of, and interest in metacomputing. They were also asked what level of effort they would be willing to put forth to learn a new system, and what tangible results they would require to justify the expended time and effort. The intent is to discover what features of grid systems are important to users and what features are not important.

In general, the unsophisticated users had little or no interest in metacomputing. This is not surprising, since if they were interested in improving performance they would have already started gravitating towards high performance computing. These users seem to be satisfied with the status quo. When pressed, some admitted that the ability to generate results more quickly would be of interest. However, this speedup must be achieved with minimal effort on the part of the user. A major concern expressed by these users was simple access to results. These users want to be able to access results on their local machines with little intervention on their part.

The intermediate and sophisticated users offered responses similar to each other's, with the primary difference being the amount of effort these groups are willing to put forth to meet their objectives. The intermediate users are generally willing to put forth a moderate amount of effort while the sophisticated users are willing to expend considerable effort to achieve their objectives. To clarify, intermediate users generally say they would be willing to spend from a few hours to a couple days on the learning curve to learn how to use a new system. Once the learning curve has been ascended, these users want the actual running of jobs to be no more difficult than before. The sophisticated users are generally willing to spend more time on the learning curve and are willing to deal with more complicated job submission protocols.

One common theme, however, is that both intermediate and sophisticated users seem very clear that they are willing to put forth this effort only if real, tangible benefits can be achieved. Members of neither group seem interested in exploring metacomputing just because it is new and interesting. These users are still most interested in their science, and will only be interested in metacomputing if it helps to improve their productivity.

There are two major themes that the intermediate and sophisticated users mention as benefits they would like to receive from a grid system. The first is improved completion time. It is important to distinguish between completion time and runtime. Most of these users have little interest in how long it actually takes their jobs to run. What they care about is how long it takes from the time they submit a job until the have access to the results.

The other major benefit cited by these users is easier file staging. These users are often using very large data sets. In these situations, more time and effort is often spent making sure the data are available to the application than in any other aspect of the job submission process. Users encountering such situations have expressed significant interest in any system that would make this file staging easier.

5. CONCLUSIONS

For the most part, there seems to be little interest in metacomputing among the end user community. While supervisors, systems support, and user support people seem to recognize the potential offered by the grid, the domain scientists have yet to notice.

This lack of interest seems to be particularly prevalent among the unsophisticated users. These users are not interested in anything that will take a significant amount of time from their science. They may be interested in systems that a very easy to learn and very easy to use. Network enabled servers or similar systems that require little code alteration and/or little programming would appear to be the types of systems that may appeal to the unsophisticated user. However, it will be very difficult to generate interest among these users for any systems that have significant learning curves or require significant additional effort to use.

Since intermediate and sophisticated users are already availing themselves of HPC resources, network enables servers and similar systems will offer little benefit to these groups. These users may be interested in systems that provide improved scheduling, leading to faster turn around time. Since it seems unlikely that any grid systems will soon allow these users to run their codes with shorter runtimes, the only way to achieve shorter turnaround time is improve scheduling techniques. These users have also expressed interest in systems that will simplify file staging. This is particularly true for users who are using very large data sets.

It must be remembered that many users are satisfied with the status quo. Even though the way they are doing things may not be the "best" way, it is something with which they are comfortable. A common response is that the way they are doing things is good enough. Some responses from both unsophisticated and intermediate users simply stated that there was nothing metacomputing could offer that would make it worth their time to start using it. These users satisfied with the status quo need significant benefits to be available.

Finally, it seems clear that more effort must be made to get the end users interested in grid computing and to better define what the users' requirements are and what the benefits will be. This paper serves as a starting point for defining users' requirements, but due to the general apathy towards grid computing displayed by the user community, it is difficult to gather comprehensive results.

Acknowledgement

The authors would like to thank Kevin Roe at the Maui High Performance Computing Center, and Tom Cortese, on-site Computational Environments lead at NAVO, for their help in gathering user opinions.

References

1. S. Agrawal. *Hardware Software Server in Netsolve*. Innovative Computing Laboratory Technical Report, ICL-UT-02-02, March 2002
2. W. Allcock, A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, ans S. Tuecke. The Data Grid: Towards an Architecture for the Distributed Management and Analysis of Large Scientific Datasets. *Journal of Network and Computer Applications (23)*, 2001
3. D. Arnold, S. Vadhiyar, and J. Dongarra. On the Convergence of Computational and Data Grids. In *Parallel Processing Letters*, Volume 11, Numbers 2-3, September 2001
4. F. Berman, et. Al. The GrADS Project: Software Support for High-level Grid Application Development. *International Journal of High Performance Computing Applications*, 15(4), 2001
5. F. Berman and R. Wolski. The AppLeS Project: A Status Report. In *Proceedings of the 8th NEC Research Symposium*, Berlin, Germany, May 1997
6. J. Bester, I. Foster, C. Kesselman, J. Tedesco, and S. Tuecke. GASS: A Data Movement and Access Service for Wide-Area Computing Systems. In *Sixth Workshop on I/O in Parallel and Distributed Systems*, Atlanta, USA, May 1999
7. H. Casanova, and J. Dongarra. NetSolve: A Network Server for Solving Computational Science Problems. *The international Journal of Supercomputing Applications and High Performance Computing*, 11(3), 1997
8. H. Casanova, et. Al. The Virtual Instrument: Support for Grid-enabled Scientific Simulations. *Journal of Parallel and Distributed Computing*, submitted April 2002
9. H. Casanova, S. Matsuoka, and J. Dongarra. Network-Enabled Server Systems: Deploying Scientific Simulations on the Grid. In *Proceedings of 2001 High Performance Computing Symposium (HPC'01), Part of the Advanced Simulation Technologies Conference*, Seattle, USA, April 2001
10. K. Czajkowski, I. Foster, C. Kesselman, S. Martin, W. Smith, and S. Tuecke. A Resource Management Architecture for Metacomputing Systems. In *Proceedings of 12th International Parallel Processing Symposium & 9th Symposium on Parallel and Distributed Processing, Workshop on Job Scheduling Strategies for Parallel Processing*, 1998, Orlando, FL.
11. I. Foster and C. Kesselman. Globus: A Metacomputing Infrastructure Toolkit. *International Journal of Supercomputing applications*, 11(2), PP 115-128.

12. I. Foster and C. Kesselman, editors. *The Grid, Blueprint for a New Computing Infrastructure.* Morgan Kaufmann Publishers, Inc., San Francisco, USA, 1998
13. I. Foster, C. Kesselman, and S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International Journal on High Performance Computing Applications*, 15(3), 2001
14. J. Gehring and A. Reinefeld. MARS – A Framework for Minimizing the Job Execution Time in a Metacomputing Environment. *Future Generation Computer Systems, FGCS 12(1)*, 1996
15. Global Grid Forum Webpage. http://www.gridforum.org.
16. Globus Webpage. http://www.globus.org.
17. Grid Portal Collaboration Webpage. http://www.ipg.nasa.gov/portals
18. GridPort Webpage. http://gridport.npaci.edu
19. A. Grimshaw, F. Ferrari, A. Knabe, and M. Humphry. Wide-Area Computing: Resource Sharing on a Large Scale. *IEEE Computer*, 32(5), 1999
20. GriPhyN Webpage. http://www.griphyn.org.
21. K. Kennedy et. al. Toward a Framework for Preparing and Executing Adaptive Grid Programs. In *Proceedings of NSF Next Generation Systems Program Workshop (International Parallel and Distributed Processing Symposium 2002*. Fort Lauderdale, USA, April 2002
22. M. Miller, C. Moulding, J. Dongarra, and C. Johnson. Grid-enabling Problem Solving Environments: A Case Study of SCIRUN and NetSolve. In *Proceedings of 2001 High Performance Computing Symposium (HPC'01), Part of the Advanced Simulation Technologies Conference*, Seattle, USA, April 2001
23. H. Nakada, M. Sato, and Sekiguchi. Design and Implementation of Ninf: Towards a Global Computing Infrastucture. *Future Generation Computing Systems, Metacomputing Issue*, 1999
24. National Virtual Collaboratory for Earthquake Engineering Research Webpage. http://www.neesgrid.org.
25. Particle Physics Data Grid Webpage. http://www.ppdg.net
26. J. Plank, M. Beck, W. Elwasif, T. Moore, M. Swany, and R. Wolski. IBP – The Internet Backplane Protocol: Storage in the Network. In *NetStore '99: Network Storage Symposium*, Seatle, USA, October 1999
27. The Purdue University Network Computing Hubs Homepage. http://www.punch.ecn.purdue.edu
28. S. Sekiguchi, M. Sato, H. Nakada, S. Matsuoka, and U. Nagashima. Ninf: Network Based Information Library for Globally High Performance Computing. In *Proceedings of Parallel Object Oriented Methods and Applications (POOMA)*, Sante Fe, USA, February 1996
29. S. Vadhiyar and J. Dongarra. A Metascheduler for the Grid. *HPDC 2002*, Submitted, 2002
30. C. Waldspurger and W. Weihl. Lottery Scheduling: Flexible Proportional-Shre Resource Management. In *First Symposium on Operating System Design and Implementation (OSDI)*, 1995
31. J. Weissman. The Interface Paradigm for Network Job Scheduling. In *Proceedings of the Heterogeneous Computing Workshop*, April 1996